ISSN: 1803-3814

DIFFERENTIAL EVOLUTION AND DETERMINISTIC CHAOTIC SERIES: A DETAILED STUDY

Roman Senkerik[1], Adam Viktorin[1], Ivan Zelinka[2], Michal Pluhacek[1], Tomas Kadavy[1], Zuzana Kominkova Oplatkova[1], Vikrant Bhateja[3], and Suresh Chandra Satapathy[4]

[1]Tomas Bata University in Zlin, Faculty of Applied Informatics
Department of Informatics and Artificial Intelligence
Nam T.G. Masaryka 5555, 760 01 Zlin, Czech Republic
senkerik@utb.cz

[2]VŠB-Technical University of Ostrava, Faculty of Electrical Engineering and Computer Science
Department of Computer Science
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
ivan.zelinka@vsb.cz

[3]Shri Ramswaroop Memorial Group of Professional Colleges (SRMGPC)
Department of Electronics and Communication Engineering
Lucknow, U.P., India
bhateja.vikrant@ieee.org

[4]Kalinga Institute of Industrial Technology
School of Computer Engineering
Bhubaneswar, Odisha, India
sureshsatapathy@ieee.org

Abstract: *This research represents a detailed insight into the modern and popular hybridization of deterministic chaotic dynamics and evolutionary computation. It is aimed at the influence of chaotic sequences on the performance of four selected Differential Evolution (DE) variants. The variants of interest were: original DE/Rand/1/ and DE/Best/1/ mutation schemes, simple parameter adaptive jDE, and the recent state of the art version SHADE. Experiments are focused on the extensive investigation of the different randomization schemes for the selection of individuals in DE algorithm driven by the nine different two-dimensional discrete deterministic chaotic systems, as the chaotic pseudo-random number generators. The performances of DE variants and their chaotic/non-chaotic versions are recorded in the one-dimensional settings of 10D and 15 test functions from the CEC 2015 benchmark, further statistically analyzed.*

Keywords: *Differential Evolution, Complex dynamics, Deterministic chaos, Population diversity, Chaotic map*

1 Introduction

This research deals with the mutual intersection of the two computational intelligence fields, which are the complex sequencing and dynamics given by the selected chaotic systems, and evolutionary computation techniques (ECT's). Together with this persistent development of metaheuristics algorithms, chaos with its properties like ergodicity, stochasticity, self-similarity, and density of periodic orbits became very popular and modern tool for improving the performance of various ECT's. The metaheuristics algorithm of the interest here is Differential Evolution (DE) [1].

Since the key operation in metaheuristic algorithms is the randomness, recent research in chaotic approach for metaheuristics mostly uses straightforwardly various chaotic maps in the place of pseudo-random number generators (PRNG). The original chaos-based approach is tightly connected with the importance of randomization within heuristics as compensation of a limited amount of search moves. This idea has been carried out in several papers describing different techniques to modify the randomization process [2], as well as the influence of randomization operations to parameter adaptation was profoundly experimentally tested in [3].

The original concept of embedding chaotic dynamics into the evolutionary/swarm algorithms as chaotic pseudo-random number generator (CPRNG) is given in [4]. Firstly, the PSO algorithm with elements of chaos was introduced as CPSO [5], followed by the initial testing of chaos embedded DE [6], DE with chaotic mutation factor (SACDE) [7], and with the deterministic chaos for the initialization (CIDE algorithm) [8]. Original inertia weight based PSO strategy driven by CPRNGs was also profoundly investigated [9]. Besides the continuous space domain, chaos-driven metaheuristic proved to be successful also in the discrete domain [10], [11]. Recently the chaos driven heuristic concept has been utilized in several swarm-based algorithms [12], [13], as well as many applications with DE [14], [15].

The focus of this research is the deeper insight into the population dynamics of the selected DE variants when the directly embedded CPRNG is driving the indices selection. Currently, DE [1] is a well-known evolutionary computation technique for continuous optimization purposes solving many difficult and complex optimization problems. Many DE variants have been recently developed with the emphasis on control parameters self-adaptivity. DE has been modified and extended several times using new proposals of versions, and the performances of different DE variants have been widely studied and compared with other ECT's. Over recent decades, DE has won most of the evolutionary algorithm competitions in the leading scientific conferences [16], as well as being applied to several applications.

The organization of this paper is following: Firstly, the motivation for this research is proposed. The next sections are focused on the description of the concept of chaos driven DE, and the experiment background. Results and conclusion follow afterward.

2 Motivation and Related Works

Even though the hybridization of chaos and ECT's is becoming very popular, many research questions remain, as to why it works, why it may be beneficial to use the chaotic sequences for pseudo-random numbers driving the selection, mutation, crossover or other processes in particular heuristics. This paper aims to help find the way to some answers through a detailed performance analysis.

This research is an extension and continuation of the previous successful experiment with the single/multi-chaos driven PSO [7] and jDE [17], where the positive influence of hidden complex dynamics for the heuristic performance has been experimentally shown. This research is also a follow up to experiments with different sampling rates applied to the chaotic sequences resulting in keeping, partially/fully removing traces of chaos [18], further it follows findings and conclusions from population diversity analyses in chaos driven DE published in [19].

Also, this paper is an extension of the work published in [20]. Here, more detailed numerical results, and graphical analyses supporting statements in conclusion section is provided. The motivation and the originality of the presented research can be summarized as follows:

- To present a comprehensive review of the chaos embedded DE, so that the reader can easily navigate between different chaotic CPRNGs and different well known DE strategies, and to see the direct comparisons of performances and deeper insight into population dynamics.
- Totally 40 versions of DE algorithm are tested here (4 variants of DE times nine chaotic versions + one original nonchaotic).
- The CEC 15 benchmark test has been utilized here. Previously, only the results for the basic set of test functions or CEC 13 test suit have been published.
- The original strategy DE/Best/ (2) has never been tested with chaotic sequences, and it is presented here in more detailed analyses.

3 Differential Evolution

This section describes the basics of original DE, jDE and SHADE strategies. The original DE [1] has four static control parameters – a number of generations G, population size NP, scaling factor F and crossover rate CR. In the evolutionary process of original DE, these four parameters remain unchanged and depend on the initial user setting. jDE and SHADE algorithms, on the other hand, adapts the F and CR parameters during the evolution. The concept of essential operations in all four selected strategies (DE/Rand/1/, DE/Best/1/, jDE and SHADE) is shown in following sections, for a detailed description on either original DE refer to [1], for jDE see [21] or for SHADE, please see [16].

3.1 Original DE Algorithm

In this research, we have used DE "Rand/1/" (1) and "Best/1/" (2) mutation strategies with the binomial crossover. Firstly, the parent selection and detailed operations for mutation strategies are described.

The parent indices (vectors) are selected by standard PRNG with uniform distribution. Mutation strategy "rand/1" uses three random parent vectors with indexes $r1$, $r2$, and $r3$, where $r1 = \mathrm{U}[1, NP]$, $r2 = \mathrm{U}[1, NP]$, $r3 = \mathrm{U}[1, NP]$ and $r1 \neq r2 \neq r3$. Mutated vector $v_{i,G}$ is obtained from three different vectors x_{r1}, x_{r2}, x_{r3} from current generation G with the help of scaling factor F as follows (1):

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r1,G} + F\left(\boldsymbol{x}_{r2,G} - \boldsymbol{x}_{r3,G}\right). \tag{1}$$

The mutation strategy "Best/1/" uses only two random parent vectors with indexes $r1$, and $r2$, and best individual solution in the current generation. The selection respects the very same rules and features as in the previous case. Mutated vector $v_{i,G}$ is obtained as follows:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{best,G} + F\left(\boldsymbol{x}_{r1,G} - \boldsymbol{x}_{r2,G}\right). \tag{2}$$

The next step is the crossover and selection. The trial vector $\boldsymbol{u}_{i,G}$ which is compared with original vector $\boldsymbol{x}_{i,G}$ is completed by crossover operation (3). CR value in original DE algorithm is static.

$$u_{j,i,G} = \begin{cases} v_{j,i,G} & \text{if } U[0,1] \leq \mathrm{CR} \text{ or } j = j_{rand} \\ x_{j,i,G} & \text{otherwise} \end{cases}. \tag{3}$$

Where j_{rand} is a randomly selected index of a feature, which has to be updated ($j_{rand} = \mathrm{U}[1, D]$), D is the dimensionality of the problem.

The vector which will be placed into the next generation G+1 is selected in the selection step. When the objective function value of the trial vector $\boldsymbol{u}_{i,G}$ is better than that of the original vector $\boldsymbol{x}_{i,G}$, the trial vector will be selected for the next population. Otherwise, the original will survive (4).

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G} & \text{if } f(\boldsymbol{u}_{i,G}) \leq f(\boldsymbol{x}_{i,G}) \\ \boldsymbol{x}_{i,G} & \text{otherwise} \end{cases}. \tag{4}$$

3.2 jDE

The generated ensemble of two control parameters F and CR is assigned to each i-th individual of the population and survives with the solution if an individual is transferred to the new generation. If the newly generated solution is not successful, i.e., the trial vector has worse fitness than the compared original active individual; the new (possibly) reinitialized control parameters values disappear together with unsuccessful solution. The both aforementioned DE control parameters may be randomly mutated with predefined probabilities τ_1 and τ_2. If the mutation condition happens, a new random value of $CR \in [0, 1]$ is generated, possibly also a new value of F which is mutated in $[F_l , F_u]$. These new control parameters are after that stored in the new population. Input parameters are typically set to $F_l = 0.1$, $F_u = 0.9$, $\tau_1 = 0.1$, and $\tau_2 = 0.1$ as originally given in [21].

3.3 SHADE

The mutation strategy used in SHADE is "current-to-*pbest*/1/" and uses four parent vectors – current i-th vector $\boldsymbol{x}_{i,G}$, vector $\boldsymbol{x}_{pbest,G}$ randomly selected from the $NP \times p$ best vectors (regarding objective function value) from current generation G. The p value is randomly generated by uniform PRNG $U[p_{min}, 0.2]$, where $p_{min} = 2/NP$. Third parent vector $\boldsymbol{x}_{r1,G}$ is randomly selected from the current generation and last parent vector $\boldsymbol{x}_{r2,G}$ is also randomly selected, but from the union of current generation G and external archive A. Also, vectors $\boldsymbol{x}_{i,G}$, $\boldsymbol{x}_{r1,G}$ and $\boldsymbol{x}_{r2,G}$ has to differ, $\boldsymbol{x}_{i,G} \neq \boldsymbol{x}_{r1,G} \neq \boldsymbol{x}_{r2,G}$. The mutated vector $\boldsymbol{v}_{i,G}$ is generated by (5).

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F_i(\boldsymbol{x}_{pbest,G} - \boldsymbol{x}_{i,G}) + F_i(\boldsymbol{x}_{r1,G} - \boldsymbol{x}_{r2,G}). \tag{5}$$

The i-th scaling factor F_i is generated from a Cauchy distribution with the location parameter $M_{F,r}$. SHADE algorithm uses the very same crossover (3) and elitism schemes (4) as canonical DE with following differences. CR value is not static, CR_i is generated from a normal distribution with a mean parameter value $M_{CR,r}$ And the elitism process uses the historical archive. For the archive and historical memories updates, details about the parameters $M_{F,r}$ and $M_{CR,r}$ due, to the limited space here, please see [16].

4 DE with Discrete Chaotic System as Driving CPRNG

The general idea of CPRNG is to replace the default PRNG with the chaotic system. As the chaotic system is a set of equations with a static start position (See Tab. 1), we created a random start position of the system, to have different start position for different experiments. Thus we are utilizing the typical feature of chaotic systems, which is extreme sensitivity to the initial conditions, popularly known as "butterfly effect." This random position is initialized with the default PRNG, as a one-off randomizer. Once the start position of the chaotic system has been obtained, the system generates the next sequence using its current position. Used approach is based on the following definition (6), where the *rndreal* represents the normalized pseudo-random value from the typical range of 0 - 1, *rndChaos* is the current output iteration of the chaotic map (x-axis), and *maxval* is the maximum value from generated chaotic series. This approach is causing so-called folding of the attractor around y-axis.

$$rndreal = \left| \frac{rndChaos}{maxval} \right|. \tag{6}$$

Following nine well known and frequently utilized discrete dissipative chaotic maps were used as the CPRNGs for all four DE strategies. With the settings as in Tab. 1, systems exhibit typical chaotic behavior [22]. Also, Fig. 1 shows the short chaotic sequences for all nine maps. These plots support the claims that due to the presence of self-similar chaotic sequences (see either significant sequencing and periodicity, or the patterns of fractals/self-similarity), the heuristic is forced to neighborhood-based selection (or alternative communication in swarms).

5 Results

For the performance comparisons in this research, the CEC 15 benchmark suite [23] was selected. The dimension D was set to 10. Every instance was repeated 51 times with the maximum number of objective function evaluations set to 100 000, i.e. ($10{,}000 \times D$). The results were recorded for all tested algorithm – four original DE strategies and their nine versions embedded with different CPRNGs. All strategies/versions used the same set of control parameters: population size $NP = 50$ and initial settings $F = 0.5$, $CR = 0.8$ (only original DE); internal SHADE variants parameter $H = 20$. Experiments were performed in the environment of *Java*; non-chaotic DE, therefore, has used the built-in *Java linear congruential pseudorandom number generator* representing traditional pseudorandom number generator in comparisons.

The simple statistical comparisons in comprehensive tables containing minimum results are given in Tables 2 - 5. Due to the limited space here, the numerical comparison of mean (median) values is not present here. Instead, the data in Tables 2 - 5 are supported by the rankings of the algorithms depicted in Fig. 2 based on the *Friedman test with Nemenyi post hoc test* calculated on the 51 runs and 15 functions of CEC2015 benchmark in 10D. The dashed line in Fig. 2 represents the Nemenyi critical distance.

Furthermore, boxplots diagrams for selected test functions are depicted in Figures 3 - 6. Those Figures, as well as Tables 2 - 5 covers all four strategies (DE/Rand/1/, DE/Best/1/, jDE and SHADE), and their ten versions (one original and nine chaotic). Detailed results discussion is in the next section.

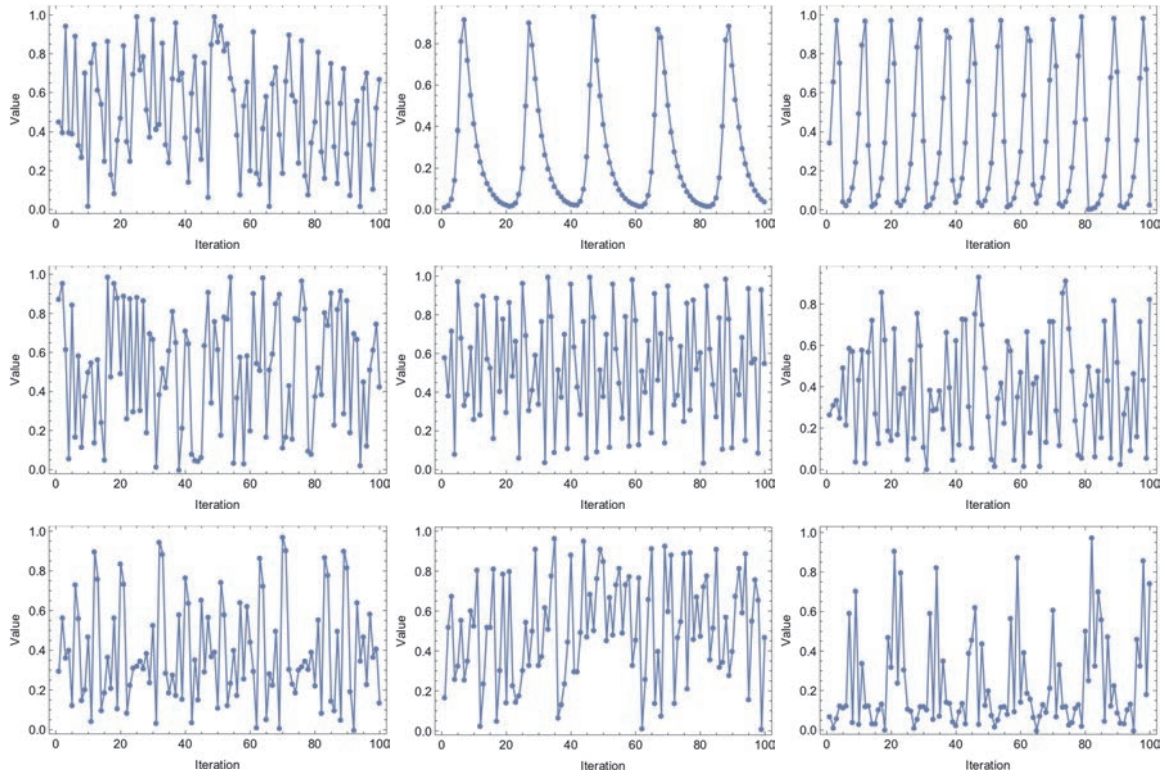

Figure 1: Chaotic sequences normalized to the typical range of 0 - 1 for CPRNG; Arnold Cat map (upper left), Burgers Map (upper middle), and so on for the other maps (Delayed Logistic, Dissipative, Henon, Ikeda, Lozi, Sinai, Tinkerbell).

Table 1: Definition of chaotic systems used as CPRNGs

Chaotic system	**Notation**	**Parameters**
Arnold Cat map	$X_{n+1} = X_n + Y_n \pmod 1$ $Y_{n+1} = X_n + kY_n \pmod 1$	$k = 2.0$
Burgers map	$X_{n+1} = aX_n - Y_n^2$ $Y_{n+1} = bY_n + X_n Y_n$	$a = 0.75$ and $b = 1.75$
Delayed Logistic	$X_{n+1} = AX_n(1 - Y_n)$ $Y_{n+1} = X_n$	$A = 2.27$
Dissipative Standard map	$X_{n+1} = X_n + Y_{n+1} \pmod{2\pi}$ $Y_{n+1} = bY_n + k \sin X_n \pmod{2\pi}$	$b = 0.1$ and $k = 8.8$
Hénon map	$x_{n+1} = a - x_n^2 + by_n$ $y_{n+1} = x_n$	$a = 1.4$ and $b = 0.3$
Ikeda map	$X_{n+1} = \gamma + \mu(X_n \cos\phi + Y_n \sin\phi)$ $Y_{n+1} = \mu(X_n \sin\phi + Y_n \cos\phi)$ $\phi = \beta - \alpha / (1 + X_n^2 + Y_n^2)$	$\alpha = 6$, $\beta = 0.4$, $\gamma = 1$ and $\mu = 0.9$
Lozi Map	$X_{n+1} = 1 - a\lvert X_n \rvert + bY_n$ $Y_{n+1} = X_n$	$a = 1.7$ and $b = 0.5$
Sinai map	$X_{n+1} = X_n + Y_n + \delta \cos 2\pi Y_n \pmod 1$ $Y_{n+1} = X_n + 2Y_n \pmod 1$	$\delta = 0.1$
Tinkerbell map	$X_{n+1} = X_n^2 - Y_n^2 + aX_n + bY_n$ $Y_{n+1} = 2X_n Y_n + cX_n + dY_n$	$a = 0.9$, $b = -0.6$, $c = 2$ and $d = 0.5$

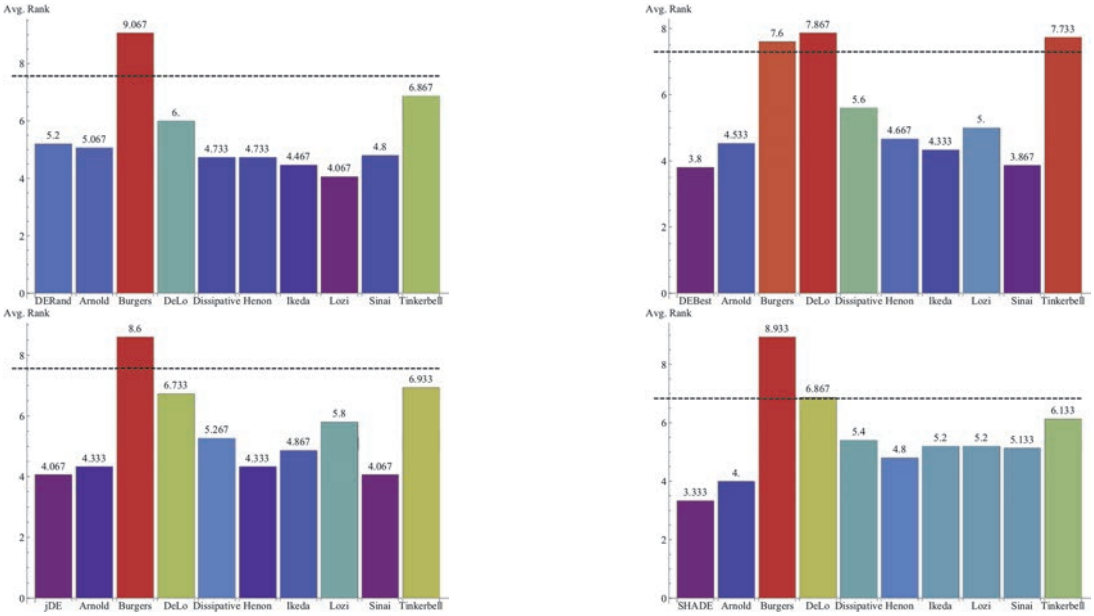

Figure 2: Ranking of algorithms: DE/Rand/ versions (upper left); DE/Best/ (upper right), jDE (bottom left); SHADE (bottom right).

Table 2: The best (minimum found) results for DERand and C_DE; CEC 2015 Benchmark set, 10*D*, 51 runs

f/system	DERand	Arnold C_DE	Burgers C_DE	DeLo C_DE	Dissipative C_DE	Henon C_DE	Ikeda C_DE	Lozi C_DE	Sinai C_DE	Tinkerbell C_DE
1	$6.82\cdot10^{-7}$	0	97.8299	0.01533	0	0	$1.00\cdot10^{-10}$	0	0	100.265
2	0	0	0	0	0	0	0	0	0	0
3	19.8787	0	19.7117	0	20.0688	11.1445	0	$4.82\cdot10^{-8}$	$1.04\cdot10^{-7}$	0
4	16.5301	3.41858	1.98996	1.98992	1.98992	$2.14\cdot10^{-3}$	1.9899	1.9899	1.5985	0.9950
5	815.004	35.1138	10.1824	0.2498	3.5399	0.3747	3.6647	3.6648	18.597	3.3525
6	0.6137	0	1.20682	0.2081	0	0	0	0	0	0.9950
7	0.2617	$1.0\cdot10^{-10}$	0.02687	0	0	0	$7.40\cdot10^{-3}$	$1.00\cdot10^{-10}$	$7.41\cdot10^{-3}$	0
8	0.3088	$2.18\cdot10^{-6}$	$9.68\cdot10^{-4}$	$9.38\cdot10^{-4}$	$6.30\cdot10^{-5}$	$7.34\cdot10^{-7}$	$3.56\cdot10^{-6}$	$7.90\cdot10^{-5}$	$9.27\cdot10^{-7}$	$4.25\cdot10^{-3}$
9	100.195	100.142	100.140	100.069	100.129	100.103	100.117	100.109	100.087	100.138
10	216.746	216.537	216.728	216.538	216.537	216.537	216.537	216.537	216.537	211.168
11	200.542	0.9309	1.87032	0.3397	0.7976	0.8104	0.6809	0.8940	0.9886	0.7443
12	101.786	101.087	100.881	100.696	100.671	101.081	100.756	100.569	101.042	100.519
13	31.0225	27.2289	27.2325	22.7848	25.9002	24.6357	25.2364	25.5163	27.0663	23.8107
14	4352.12	2935.54	2935.54	100	2935.54	100	100	2935.54	2935.54	100
15	100	100	100	100	100	100	100	100	100	100

Table 3: The best (minimum found) results for DEBest and C_DE; CEC 2015 Benchmark set, 10*D*, 51 runs

f/system	DEBest	Arnold C_DE	Burgers C_DE	DeLo C_DE	Dissipative C_DE	Henon C_DE	Ikeda C_DE	Lozi C_DE	Sinai C_DE	Tinkerbell C_DE
1	0	0	0	0	0	0	0	0	0	0
2	11119.4	0	0	0	0	0	0	0	0	0
3	20.174	20.0144	$2.04\cdot10^{-5}$	20.0282	2.5799	20.0478	7.350	0	20.0286	2.0161
4	18.3524	3.9798	6.2651	5.9698	5.9698	3.9798	5.9698	6.9647	4.9750	1.9899
5	491.466	6.8359	151.049	122.122	120.13	33.4043	6.8924	15.3694	33.4668	25.2442
6	388.906	44.9809	13.1426	13.3507	25.8688	17.3305	36.2548	5.3911	14.1578	128.596
7	2.3578	1.0630	0.2192	0.7505	0.4896	1.0331	0.1390	0.6661	0.1181	0.3245
8	168.463	0.3260	0.8083	0.6296	0.4801	0.8156	0.4994	0.8325	0.3202	1.0129
9	100.35	100.222	100.157	100.209	100.156	100.178	100.12	100.109	100.125	100.206
10	367.638	217.122	204.423	191.54	217.015	198.852	195.978	194.098	195.995	221.519
11	296.3	7.6740	9.4889	9.5429	2.9227	3.8901	2.4784	4.6306	4.5402	5.50815
12	103.012	101.375	101.476	101.176	101.041	101.508	101.535	101.368	101.698	101.607
13	38.1593	26.8363	26.3723	28.115	26.6198	28.361	28.6205	27.9272	21.6638	29.2378
14	5588.34	100	101.873	104.979	1662.67	100.002	100	100	100	118.161
15	101.002	100	100	100	100	100	100	100	100	100

Table 4: The best (minimum found) results for jDE and C_jDE; CEC 2015 Benchmark set, 10*D*, 51 runs

f/system	jDE	Arnold C_jDE	Burgers C_jDE	DeLo C_jDE	Dissipative C_jDE	Henon C_jDE	Ikeda C_jDE	Lozi C_jDE	Sinai C_jDE	Tinkerbell C_jDE
1	$9.79\cdot10^{-8}$	0	$7.01\cdot10^{-5}$	$4.92\cdot10^{-7}$	0	0	0	0	0	$3.15\cdot10^{-4}$
2	0	0	0	0	0	0	0	0	0	0
3	19.7916	$2.50.10^{-9}$	0.1154	5.5140	6.5179	$2.39\cdot10^{-4}$	11.9019	6.4830	$3.25\cdot10^{-4}$	7.3023
4	4.1515	1.2559	0	0	1.1008	1.2774	1.8199	1.5781	1.1066	1.0075
5	126.703	37.1301	10.3074	25.7866	37.3398	14.2125	40.4215	33.1746	23.3597	33.8032
6	6.1719	0	0.002150	0.2081	0	0	0	0	0	0
7	0.1724	0.03687	0.01968	0.03108	0.04643	0.05496	0.03811	0.04974	0.03340	$9.35\cdot10^{-3}$
8	0.2298	$7.11\cdot10^{-7}$	$6.36\cdot10^{-6}$	$3.69\cdot10^{-6}$	$1.69\cdot10^{-5}$	$9.69\cdot10^{-5}$	$3.29\cdot10^{-6}$	$3.38\cdot10^{-4}$	$3.38\cdot10^{-7}$	$1.30\cdot10^{-4}$
9	100.207	100.152	100.154	100.121	100.108	100.134	100.121	100.129	100.126	100.148
10	218.286	216.537	216.556	216.537	216.537	216.537	216.537	216.537	216.537	216.539
11	101.593	0.4639	1.6532	1.0906	0.7982	0.9954	0.8272	0.4429	0.5749	1.15227
12	101.839	101.245	101.308	100.994	100.948	100.928	100.801	101.238	101.067	100.409
13	27.961	24.0059	21.7485	23.3842	23.8183	23.6141	24.4654	25.8824	23.7317	23.0729
14	3613.67	2935.54	2935.54	2935.54	2935.54	100	100	2935.54	2935.54	100
15	100	100	100	100	100	100	100	100	100	100

Table 5: The best (minimum found) results for SHADE and C_SHADE; CEC 2015 Benchmark set, 10*D*, 51 runs

f/system	SHADE	Arnold C_SHADE	Burgers C_SHADE	DeLo C_SHADE	Dissipative C_SHADE	Henon C_SHADE	Ikeda C_SHADE	Lozi C_SHADE	Sinai C_SHADE	Tinkerbell C_SHADE
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	17.8977	0.5591	1.8283	2.4405	2.0426	2.4463	1.2702	2.3933	2.2275	2.0514
4	3.2186	1.9906	0.9981	0.9958	1.9914	$2.35\cdot10^{-3}$	0.9964	$2.46\cdot10^{-3}$	0.04048	$1.80\cdot10^{-3}$
5	83.2657	6.8927	3.7233	1.0436	3.7805	0.5567	1.6673	1.2354	3.7573	1.1339
6	9.0269	0	0.416286	0	0	0	0	0	0	0
7	0.2429	0.03910	0.01971	0.03654	0.03797	0.02615	0.03442	0.01782	0.03105	$4.24\cdot10^{-3}$
8	0.4981	$6.41\cdot10^{-4}$	0.01371	$2.98\cdot10^{-6}$	$1.85\cdot10^{-4}$	$9.64\cdot10^{-4}$	$1.85\cdot10^{-3}$	$4.73\cdot10^{-5}$	$1.48\cdot10^{-4}$	0.09180
9	100.21	100.124	100.117	100.115	100.123	100.085	100.117	100.122	100.135	100.114
10	217.657	216.537	100.	216.537	216.537	216.537	216.537	216.537	216.537	216.537
11	213.029	1.7574	2.2380	2.7001	0.7395	2.13053	2.4767	2.2032	2.2932	2.3281
12	101.473	100.911	100.886	100.923	100.92	101.095	100.963	100.875	100.881	100.797
13	28.4103	24.8635	25.2288	24.8443	22.5237	25.9536	22.8663	23.001	23.6385	24.1002
14	3641.88	100	100	100	100	100	2935.54	100	100	100
15	100	100	100	100	100	100	100	100	100	100

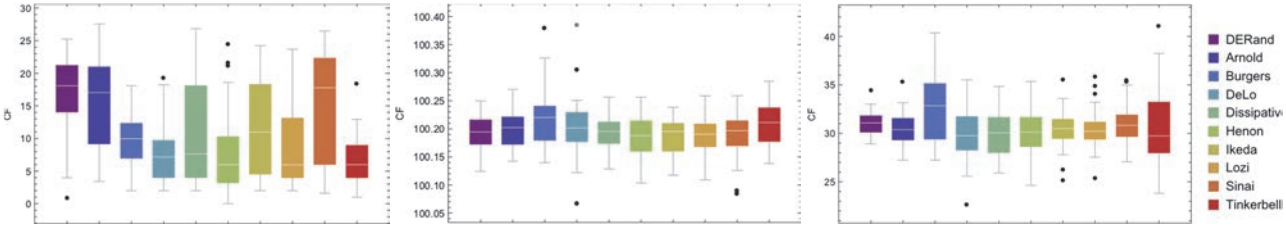

Figure 3: Boxplots for DE/Rand versions and selected functions in 10*D*, 51 runs; from left: *f*4, *f*9 and *f*13.

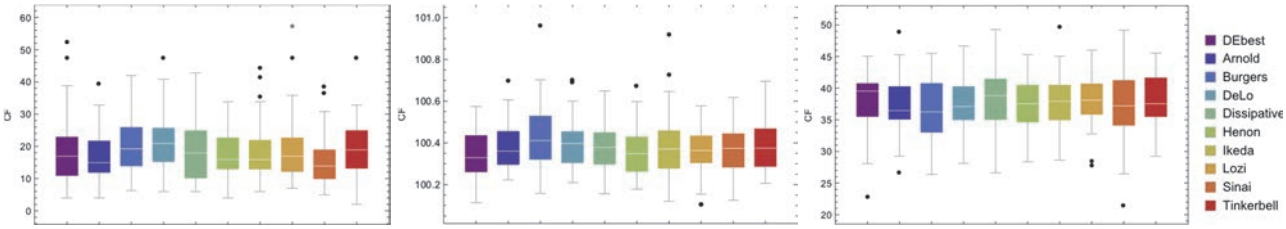

Figure 4: Boxplots for DE/Best versions and selected functions in 10*D*, 51 runs; from left: *f*4, *f*9 and *f*13.

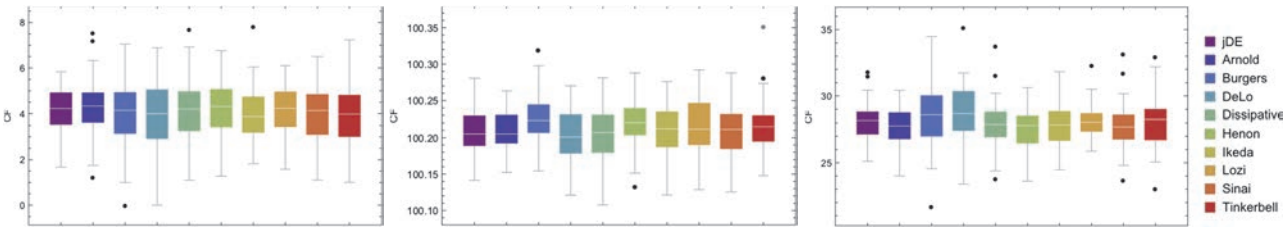

Figure 5: Boxplots for jDE versions and selected functions in 10*D*, 51 runs; from left: *f*4, *f*9 and *f*13.

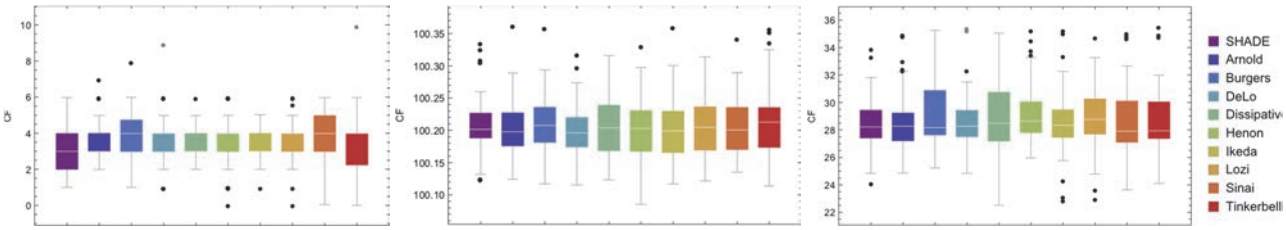

Figure 6: Boxplots for SHADE versions and selected functions in 10*D*, 51 runs; from left: *f*4, *f*9 and *f*13.

6 Results Analysis

The findings can be summarized as:

- Obtained graphical rankings (see Fig.2) and the min. (best) obtained objective function values given in Tables 2 - 5, support the claim that all studied four DE strategies are sensitive to the chaotic dynamics driving the selection (mutation) process through CPRNG. At the same time, it is clear that (selection of) the best CPRNGs are problem-dependent. By using the CPRNG inside the heuristic, its performance is (significantly) different: either better or worse against other compared versions.
- Overall, chaotic versions seem to be very effective regarding finding the min. values of the objective function (See Tables 2 - 5 and boxplots depicted in Figures 3 - 6).
- Regarding the dimension of the problem being solved, based on our previous research, it may be claimed, that with the increase of the dimension, the influence of the unconventional randomization is strengthened. Here, we have presented only the one case of $D = 10$, which may be close to the simpler real engineering problems.
- The statistical performance comparisons (rankings) in Fig. 2 reveal the fact that original full random strategy "Rand/1/" seems to be the best choice for the hybridization with chaos. The "Best/1/" strategy shows the possible conflict between the attraction to the "best" solution in the population and other indices selected based on the chaotic series. The similar effect can be observed from the SHADE variants final rankings (Fig. 5; bottom right), where the chaotic sequencing for indices selection may be suppressed by the operations with external archive and the structure of "current-to-pbest/1" strategy. The simple parameter adjustment jDE version joining fully randomized "Rand/1/" strategy and simple adaptation of control parameters seems to be the conservative choice for optimization experiments hybridizing adaptive DE and chaos. Overall, the parameter adaptation is beneficial, regarding the improvements in finding the min. values as in Tables 2 - 5. However, it is necessary to keep in mind, that inner parameters of the chaotic maps, have not been adjusted during the DE run.
- Mostly the performance of compared original DE and chaotic DE variants is similar, or in some instances, the chaotic versions performed significantly worse. Such a worse performance was repeatedly observed for three chaotic maps: Delayed Logistic, Burgers and Tinkerbell. On the other hand, these maps usually secured robust progress towards function extreme (local) followed by premature population stagnation phase. The following chaotic maps have given the stable performance: Arnold cat, Sinai, Henon, and Ikeda.
- All discussed findings mentioned above support the theory that unique features of the chaos transformed into the sequencing of CPRNG values may increase the diversity of the population in the initial phases of the optimization. Further, the chaotic series may create the subpopulations (or inner neighborhood selection schemes). Thus, the metaheuristic can benefit from the prolonged exploration, searching within those sub-populations and quasi-periodic exchanges of information between individuals.

7 Conclusion

The primary aim of this original work is to provide a more in-depth insight into the inner dynamics of indices selection in DE. The focus is to experimentally investigate the influence of different types of unconventional non-random (chaotic) sequences to the performance of different classes of DE strategies.

The research of randomization issues and insights into the inner dynamic of metaheuristic algorithms was many times addressed as essential and beneficial. The results presented here support the approach for multi-chaotic generators [24] or ensemble systems, where we can profit from the combined/selective population diversity (i.e., exploration/exploitation) tendencies, sequencing-based either stronger or moderate progress towards the function extreme, all given by the smart combination of multi-randomization schemes.

However, a lot of analyses and different scenarios for the future research are required to support the facts mentioned above fully and to provide deeper insight into the metaheuristic dynamics, its ergodicity, complexity, and sensitivity to the unconventional randomization for particular processes.

Acknowledgement: This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic within the National Sustainability Programme Project no. LO1303 (MSMT-7778/2014), further by the European Regional Development Fund under the Project CEBIA-Tech no. CZ.1.05/2.1.00/03.0089 and by Internal Grant Agency of Tomas Bata University under the Projects no. IGA/CebiaTech/2018/003. This work is also based upon support by COST (European Cooperation in Science & Technology) under Action CA15140, Improving Applicability of Nature-Inspired

Optimisation by Joining Theory and Practice (ImAppNIO), and Action IC1406, High-Performance Modelling, and Simulation for Big Data Applications (cHiPSet). The work was further supported by resources of A.I.Lab at the Faculty of Applied Informatics, Tomas Bata University in Zlin (ailab.fai.utb.cz). Author Ivan Zelinka acknowledges SGS Grant of VSB-Technical University of Ostrava: No. SGS 2018/177.

References

[1] Das, S., Mullick, S.S., Suganthan, P.N.: Recent advances in differential evolution–an updated survey. Swarm and Evolutionary Computation 27, 1–30 (2016).
[2] Neri, F., Iacca, G., Mininno, E.: Disturbed exploitation compact Differential Evolution for limited memory optimization problems. Information Sciences **181**(12), 2469–2487 (2011).
[3] Zamuda, A., Brest, J.: Self-adaptive control parameters′ randomization frequency and propagations in differential evolution. Swarm and Evolutionary Computation 25, 72–99 (2015).
[4] Caponetto, R., Fortuna, L, Fazzino, S., Xibilia, M.G.: Chaotic sequences to improve the performance of evolutionary algorithms. IEEE Transactions on Evolutionary Computation **7**(3), 289–304 (2003).
[5] Coelho, L.d.S., Mariani, V.C.: A novel chaotic particle swarm optimization approach using Hénon map and implicit filtering local search for economic load dispatch. Chaos, Solitons & Fractals **39**(2), 510–518 (2009).
[6] Davendra, D., Zelinka, I., Senkerik, R.: Chaos driven evolutionary algorithms for the task of PID control. Computers & Mathematics with Applications **60**(4), 1088–1104 (2010).
[7] Zhenyu, G., Bo, C., Min, Y., Binggang, C.: Self-adaptive chaos differential evolution. In: International Conference on Natural Computation, pp. 972–975. Springer, Berlin, Heidelberg (2009).
[8] Ozer, A.B.: CIDE: chaotically initialized differential evolution. Expert Systems with Applications **37**(6), 4632–4641, (2010).
[9] Pluhacek, M., Senkerik, R., Davendra, D.: Chaos particle swarm optimization with ensemble of chaotic systems. Swarm and Evolutionary Computation 25, 29–35 (2015).
[10] Metlicka, M., Davendra, D.: Chaos driven discrete artificial bee algorithm for location and assignment optimisation problems. Swarm and Evolutionary Computation 25, 15–28 (2015).
[11] Davendra, D., Bialic-Davendra, M., Senkerik, R.: Scheduling the Lot-Streaming Flowshop scheduling problem with setup time with the chaos-induced Enhanced Differential Evolution. In: 2013 IEEE Symposium on Differential Evolution (SDE), pp. 119–126. IEEE.
[12] Gandomi, A.H., Yang, X.S., Talatahari, S., Alavi, A.H.: Firefly algorithm with chaos. Communications in Nonlinear Science and Numerical Simulation **18**(1), 89–98 (2013).
[13] Wang, G.G., Deb, S., Gandomi, A.H., Zhang, Z., Alavi, A.H.: Chaotic cuckoo search. Soft Computing **20**(9), 3349–3362 (2016).
[14] Coelho, L.d.S., Ayala, H.V.H., Mariani, V.C.: A self-adaptive chaotic differential evolution algorithm using gamma distribution for unconstrained global optimization. Applied Mathematics and Computation **234**(0), 452–459 (2014).
[15] Coelho, L.d.S., Pessoa, M.W.: A tuning strategy for multivariable PI and PID controllers using differential evolution combined with chaotic Zaslavskii map. Expert Systems with Applications **38**(11), 13694–13701 (2011).
[16] Tanabe, R., Fukunaga, A.S.: Improving the search performance of SHADE using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 1658–1665. IEEE (2014).
[17] Senkerik, R., Pluhacek, M., Zelinka, I., Viktorin, A., Oplatkova, Z.K.: Hybridization of Multi-chaotic Dynamics and Adaptive Control Parameter Adjusting jDE Strategy. In: International Conference on Soft Computing-MENDEL, pp. 77–87. Springer, (2016).
[18] Senkerik, R., Pluhacek, M., Zelinka, I., Davendra, D., Janostik, J.: Preliminary Study on the Randomization and Sequencing for the Chaos Embedded Heuristic. In: Abraham A, Wegrzyn-Wolska K, Hassanien EA, Snasel V, Alimi MA (ed.) Proceedings of the Second International Afro-European Conference for Industrial Advancement AECIA 2015, pp 591–601. Springer International Publishing, (2016).
[19] Senkerik, R., Viktorin, A., Pluhacek, M., Kadavy, T.: On the Population Diversity for the Chaotic Differential Evolution. In: 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2018).
[20] Senkerik, R., Viktorin, A., Pluhacek, M., Kadavy, T., Oplatkova, Z.K.: Differential Evolution and Chaotic Series. In: 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 1–5. IEEE (2018).
[21] Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE transactions on evolutionary computation **10**(6), 646–657 (2006).
[22] Sprott, J.C.: *Chaos and Time-Series Analysis*. Oxford University Press (2003).
[23] Chen, Q., Liu, B., Zhang, Q., Liang, J.J., Suganthan, P.N., Qu, B.Y.: Problem Definition and Evaluation Criteria for CEC 2015 Special Session and Competition on Bound Constrained Single-Objective Computationally Expensive Numerical Optimization. Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, (2014).
[24] Viktorin, A., Pluhacek, M., Senkerik, R.: Success-history based adaptive differential evolution algorithm with multi-chaotic framework for parent selection performance on CEC2014 benchmark set. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 4797–4803. IEEE (2016).